Fall 2012, Test 1, Introduction to Algorithms

Name:
Section:
Email id:

29th September, 2014

Answer all Six questions. You have 90 minutes to complete the exam.
You can use a calculator - You are allowed to use your text books and notes..

1. **Recursion and Design and Analysis**

(a) Write an algorithm (C++ like program) to find the smallest integer greater than or equal to $\sqrt{(n)}$ of a given input positive integer, n. (You are not allowed to use built in mathematical libraries.. You are allowed to use any of the standard (+,-*,/) arithmetical operators) For example if $n = 3$, the output should be 2, if $n = 16$, the output should be 4 and if $n = 69$, the output should be 9. What is the running time of your algorithm in O notation. [5 **points**]

```
i = 1;
while (i*i < n)
{   i = i+1;
}
return i;
```

$O(\sqrt{n})$

(b) Given a set, S, of n integer elements (e.g., $n = 6$ and $S = \{-2, -92, 12, 4, -15, -3\}$, the numbers -2,-2,4 add up to 0.), write an algorithm to find whether any three elements (not necessarily distinct) add up to 0. [A pseudo code will suffice] What is the complexity of your algorithm. [5 **points**]

```
Sort S;          S is a sorted array
for i = 0 to n-1
    a = S[i]
    j = i
    k = n-1
    while ( k >= j )
    {
        sum = a + S[j] + S[k] ;
            if   sum == 0    return (a, S[j], S[k])
            if   sum < 0     j = j+1
                             else k = k-1
    }
return ("not found");
```

$O(n^2)$

2. **Recurrence Relations, Master Theorem**

(a) Match the following Recurrence Relations with the solutions given below. (give tight bounds)[**5 Points**]

i. $P(n) = 27P(\frac{n}{3}) + 2,\ P(1) = 1$ $O(n^3)$

ii. $Q(n) = Q(n-1) + n, Q(1) = 1$ $O(n^2)$

iii. $R(n) = R(\frac{n}{2}) + \frac{1}{2}, R(1) = 1$ $O(\log n)$

iv. $S(n) = 4S(\frac{n}{4}) + n,\ S(1) = 1$ $O(n \log n)$

v. $T(n) = T(n/2) + n, T(1) = 1$ $O(n)$

Solutions

i. $O(\log(n))$ R(S)

ii. $O(n^3 \log(n))$

iii. $O(n)$ T(n)

iv. $O(n^2)$ Q(n)

v. $O(n^3)$ P(n)

vi. $O(n \log(n))$ S(n)

(b) How many print statements (cout) will be execute by the following program when n = 2, n = 3 and n = 7. Write a recurrence relation (not solve) for the number of times print statements are executed for a general n. [**5 Points**]

$T(2) = 5,\ T(3) = 5;\ T(7) = 29$

```
void csci2300(int n)
{
  if (n<=1) cout << ''Scientia et Instantia'' << endl;
  else
  {
    cout << ''Knowledge and Thoroughness'' << endl;
    csci2300(n-2);
    cout << ''Why not Change the World'' << endl;
    csci2300(n-2);
    cout << ''Let us Go Red'' << endl;
  }
return;
}
```

$T(0) = 1$

$T(1) = 1$

$T(n) = 2T(n-2) + 3$

$T(2) = 5$

$T(3) = 5$

$T(4) = 13$

$T(5) = 13$

$T(6) = 29$

$T(7) = 29$

Main idea:
compare middle element with its neighbors if it exists.
if middle element is not peak and its left neighbor is greater then there is a peak in left component.

3. **Divide and Conquer and Design of Algorithm**

Given an array of integers. Find a peak element in it. An array element is peak if it is NOT smaller than its neighbors. For corner elements, we need to consider only one neighbor. For example, for input array {15, 16, 22, 18}, 22 is the only peak element. For input array {11, 21, 16, 3, 24, 91, 68}, there are two peak elements: 21 and 91. Note that we need to return any one peak element. A linear time algorithm will get you only 5 points.

(a) Provide a pseudo code description of an algorithm to find the peak element. [**5 points**]

(b) What is the complexity of your algorithm. [**5 points**]

There will always be a peak. (binary search)

```
int findPeakUtil (int arr[], int low, int high, int n)
{ int mid = low + (high - low)/2 ;
  if (mid == 0) || arr[mid-1] <= arr[mid])
     && (mid == n-1 || arr[mid+1] <= arr[mid]))
    return arr[mid];
  else if (mid > 0 && arr[mid-1] > arr[mid]
      return findPeakUtil(arr, low, (mid-1), n);
  else
      return findPeakUtil(arr, (mid+1), high, n);
}
int findPeak (int arr[], int n)
{ retun findPeakUtil (arr, 0, n-1, n);
}
```


O(log n)

if middle element is not peak and its right neighbor is greater then peak will be in the right component.

$1 = 26*87 - 119*19$

inverse of 19 mod 87 = -119

$= -32$

$= 55$

4. **Algorithms with Numbers**

   (a) Find x and y such that $87 \times x + 19 \times y = gcd(87, 19)$ and use it compute multiplicative inverse of 19 mod (87) [**6 points**]

a	b	quot	rem
87	19	4	11
19	11	1	8
11	8	1	3
8	3	2	2
3	2	1	1
2	1	2	0
1	0		

$1 = 1 - 0$
$= 1-(2-2*1)$
$= 3*1 - 2$
$= 3*(3-1*2) - 2$
$= 3*3 - 4*2$
$= 3*3 - 4*(8-2*3)$
$= 11*3 - 4*8$
$= 11*(11-1*8) - 4*8$
$= 11*11 - 15*8$
$= 11*11 - 15*(19-1*11)$
$= 26*11 - 15*19$
$= 26*(87-4*19) - 15*19$
$= 26*87 - 119*19$

   (b) what is the value of $3^{8^{128}}$ *mod* 85 (Hint: 85 is a product of two prime numbers 5 and 17). You have to show your work (just writing the answer gives 2 points) and what theorems/lemmas you used. [**4 points**]

$3^{8^{128}}$ mod 85

3^{64} mod 85 $\equiv 1$ as $85 = 5*17$

$a^{(p-1)(q-1)}$ mod $pq = 1$

$3^{8^{2*64}}$ mod 85

$= 3^{64^{64}}$ mod 85

$= 1$

5. **Recursive full binary Trees**

A full binary tree is a tree that has either 0 or 2 children (both left and right) at each vertex .(This is **a Full Binary tree**). Let B_n denote the number of full Binary trees with n vertices. Naturally B_1=1. By definition $B_2 = 0$. In fact the number of full Binary Trees with even number of nodes is 0. i.e., $B_{2n} = 0$.

(a) Draw Binary trees with 5 and 7 vertices, determine the exact values of B_5, B_7 [**6 points**].

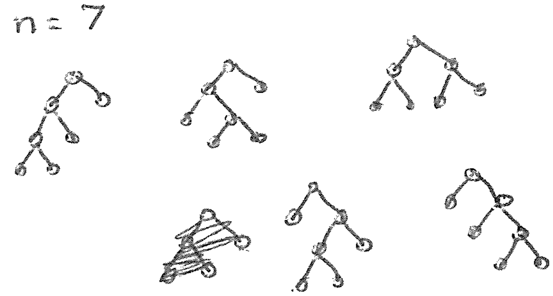
(b) For general $2n + 1$, derive a recurrence relation (and not solve) for B_{2n+1} [**4 points**] (Hint: There are $2i$+1 nodes in the left subtree,$2n - 1 - 2i$ nodes in the right subtree)

(a)

$n=5$

(2)

$n=7$



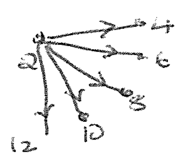
(5)

(b)

$$B_{2n+1} = \sum_{i=0}^{n-1} B(2i+1)\, B(2n-1-2i)$$

6. **Graphs and Graph Algorithms**

(a) For a directed graph G with node set =2,3,4,5,6,7,8,9,10,11,12 with a directed edge from node i to node j, if i properly (remainder is zero) divides j. (By definition there are no edges from i to i - it is not allowed for this problem). What are the out-degrees of node 2 and node 4. What are the indegee of node 12 and node 11. List all nodes with in-degree 0. [**6 points**]



outdegree (2) = 5

outdegree (4) = 2

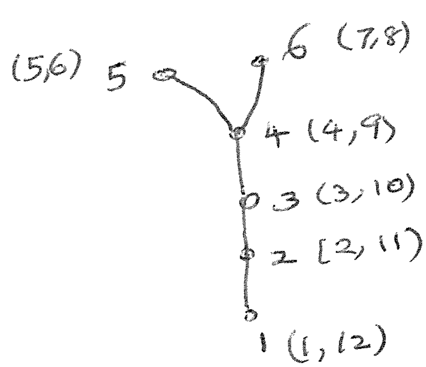
indegree of 12 = |{2, 3, 4, 6}| = 4

indegree of 11 = 0

indegree 0

2, 3, 5, 7, 11

(b) Construct a graph G, with 6 nodes and 5 edges such that the pre and post numbers for all but two of the nodes are differ by at least 5 (i.e. for 4 of the 6 nodes , post-numbering is at least 5 more than the pre-numbering. i.e, for a node n, post(n) > pre(n)+4). (You can choose a starting node and the order of traversal - show which is your starting node and the order of traversal) [**4 points**].



Nodes 1, 2, 3 and 4 satisfy this property.